Scalable Optical Tracking for Navigating Large Virtual Environments using Spatially Encoded Markers

A Practical and Accurate Low-cost Solution using Ceiling Mounted LED Strips

ABSTRACT

In this paper we present a novel approach for tracking the movement of a user in a large indoor environment. Many studies show that natural walking in virtual environments increases the feeling of immersion by the users. However, most tracking systems suffer from a limited working area or are expensive to scale up to a reasonable size for navigation.

Our system is designed to be easily scalable both in working area and number of simultaneous users using inexpensive off-the-shelf components. To accomplish this, the system determines the 6 DOF pose using passive LED strips, mounted to the ceiling, which are spatially encoded using De Bruijn codes. A camera mounted to the head of the user records these patterns. The camera can determine its own pose independently, so no restriction on the number of tracked objects is required. The system is accurate to a few millimeters in location and less than a degree in orientation. The accuracy of the tracker is furthermore independent of the size of the working area which makes it scalable to enormous installations. To provide a realistic feeling of immersion, the system is developed to be real-time and is only limited by the framerate of the camera, currently at 60Hz.

Keywords

Optical tracking, real walking, wide-area, large virtual environment, low-cost

1. INTRODUCTION

Navigation is the most common interactive task performed in a large three-dimensional virtual environment (VE). Especially in immersive VEs, building an intuitive way of navigating without losing the sense of immersion is not trivial. The reason for this is that navigating in the real world is not only visual. It has been shown that there are a lot of benefits of using a walking interface to explore a virtual environment; users have a higher sense of presence compared to other locomotion techniques[22], better spatial orientation[4], have fewer collisions in the virtual world [20] and perform better on search tasks[18]. Using natural walking as an interface requires a way of tracking the position and orientation of the user over a large area. However, most tracking systems are not designed to scale up without loss of accuracy or are not cost effective when doing so.

Our system handles tracking in large areas by using a head-mounted camera and a grid of lights applied to the ceiling. Some lights in the grid are disabled, resulting in a binary pattern on the grid. By using De Bruijn codes in combination with a Manchester encoding, every pattern of 60 lights (on or off) encodes a unique location in the grid. By determining the pattern in the image of the camera, the location and orientation of the camera under the grid can be determined and global tracking can be achieved. We acquire an accuracy with a maximum error of a few millimeters for the tracking of the global location and less than one degree in tracking the global orientation. The pattern allows the use of a grid of 4.8 million km^2, without loss of accuracy. We present a prototype using off-the-shelf hardware running at 60Hz (i.e. the camera framerate), proving the method to be fast and cost effective. Because of the static nature of the grid, the drift as seen in other tracking systems is not present. This allows the simultaneous use of multiple users with correct relative location and removes the requirement of calibration while using the setup.

By allowing global and precise tracking in a large environments, other well-known techniques become available with high quality. One of these is redirected walking [25, 15, 14, 19], which reduces the physical space required for navigating large virtual environments. Here, a user can be guided to a different physical location than that he perceives in the virtual world. The technique makes use of the limits of our perception of space, as it turns out that we mainly trust our visual system. By slowly and continuously amplifying or diminishing a component of the user's motion, the user can be steered away from physical boundaries and obstacles. By interrupting or distracting the user, a smaller physical space is required [15, 25]. In terms of tracking, redirected walking also requires a larger tracking system to be effective. It also requires the actual physical location of the user to steer him/her away from the physical boundaries or other obstacles. Furthermore, interaction between different users of the virtual environment requires a correct relative position. Tracking systems who suffer from drift would become unreliable over time, making redirected walking ineffecteve.

2. RELATED TRACKING SYSTEMS

From the early creation of immersive virtual reality and the introduction of head-mounted displays (HMDs), the need to track it's position and orientation became necessary. The first systems used a mechanical linkage to accomplish such a task [21], but this confined the movement of the user to the size of the device. Magnetic-based systems gave the user more freedom to move around, but still are not suitable for larger systems, due to the inherent sensitivity to other metal and magnetic sources in the area. Therefore, using multiple base stations to create a bigger working area is not recommended. Acoustic systems on the other hand use ultrasonic sound waves to triangulate the position of the receiver. They can be scaled in a cost-effective way, but suffer from a limited and changing accuracy depending on environment conditions. Ultra-wide band (UWB) systems, like Ubisense [3], can be used over a large area. They make use of ultra-wide band radio frequency for position tracking, but are not accurate enough on their own to be used for navigating virtual environments.

When accurate tracking is required over a large - and in most cases unknown - area, an inertial system is usually used. Inertial tracking systems use inertia to sense a change in position and orientation by measuring the acceleration and torque. No other external sources or markings in the environment need to be used, which makes it not restricted to any working area. However, this also means that the tracking system has no perception about its physical location or orientation and the measured position quickly drifts from the real position. Inertial tracking systems are often combined with vision systems to counteract the weak points of each other. An example of such a hybrid tracker is the VIS-tracker [7, 27]. This tracking system uses paper patterns for absolute reference to counter drift from its inertial sensor. Because they use paper markers, their vision system is dependent on environment lighting and therefore suffers from motion blur. The patterns also need to be calibrated before the system can be used. Another system proposed by Bleser et al. [2] uses a model of the environment to find its pose. It suffers from the same limitations as the VIS-tracker, but does not require any modifications to the working area.

Another global tracking method is the well-known Global Positioning System (GPS) [9] uses a satellite-based triangulation method. The triangulation uses the differences between timestamps transmitted by the satellites, together with the location of the satellites at that time. Because the receiver only uses time differences, clock synchronization is only required between satellites. While the method is very accurate in theory, the timestamps are artificially modified to reduce accuracy to up to 5 meters of error [26], which makes GPS unsuitable for accurate global navigation in virtual environments. Furthermore, the effectiveness of the system is strongly determined by the number of visible satellites. This can introduce loss of accuracy or operation in indoor situations. Lastly, GPS does not provide orientation information, making it less suited for virtual reality applications.

Optical tracking systems use light to track the pose of an object. Most systems use an outside-looking in approach, which means that the sensors are located fixed in the world and markers are attached to the object. Most commercial systems, like Vicon, PPTX Tracker and iotracker[16], take this approach because it provides a good position accuracy of each marker by triangulation. This makes it ideal for motion capturing, but requires special 3D markers to estimate orientation. This also limits the scalability of the system because accuracy drops linearly with the distance to the sensors. Orientation magnifies this error because it is dependent on it. Furthermore, these systems can only support a limited number of users due to their design. Another drawback for immersive virtual reality is the fact that the pose of the user needs to be calculated at a distance and sent over, which introduces more latency. Building a larger working area can become costly in terms of cameras.

The HiBall system by Welch et al. [23] was especially designed for wide-area tracking. They use an inside-looking-out approach to estimate the pose of a special optical sensor. The system uses arrays of flashing infra-red LEDs which are synchronized with the sensor. The system achieves accurate 6 DOF tracking at 2000 Hz using a single-constraint-at-a-time or SCAAT algorithm [24]. Unfortunately, they can only support up to 4 sensors, because each extra sensor reduces the framerate by half. The use of special hardware can make the system expensive in larger systems.

Maesen et al. [13] introduced a low-cost scalable tracking system using inexpensive LED ropes and a head-mounted camera. No restriction on the working area or number of users was imposed, but they did not achieve a global positioning system to get the actual physical location of the user. There was no encoding of the LED lights, thus global position could not be recovered. By using temporal information, users could be tracked by differentiating positions between frames, but this is highly sensitive to frame drops. They did, however, acquire accurate orientation by using vanishing points.

Ramesh et Al. [17] present a system for motion tracking using infrared LED markers and a low-cost photodiode. The LED markers use a spatiotemporal encoding to reduce the number of LEDs. The camera is placed in the world, which implies that large area tracking is less scalable. There is a limited coverage of the scene and the distance to the markers is limited by the absence of optical lenses. However, the system is highly portable, making it practical for specific large area applications, such as movie studios.

3. OUR APPROACH

We propose a tracking system for virtual reality setups, where every user is equipped with a head-mounted display. Our system uses a head-mounted camera, directed to the ceiling. We designed our system around the concept of being scalable, which meant that we would prefer an inside-looking-out [1] approach. It is also more cost-effective when building larger systems due to the higher cost of the image sensors.

The ceiling is covered with a pattern of lights, which can be seen by the camera. The lights are placed in a grid, where some positions in the grid are disabled, i.e. no light. The markers, i.e. the (absence of) lights, are placed on the ceiling because of the relatively constant vertical distance

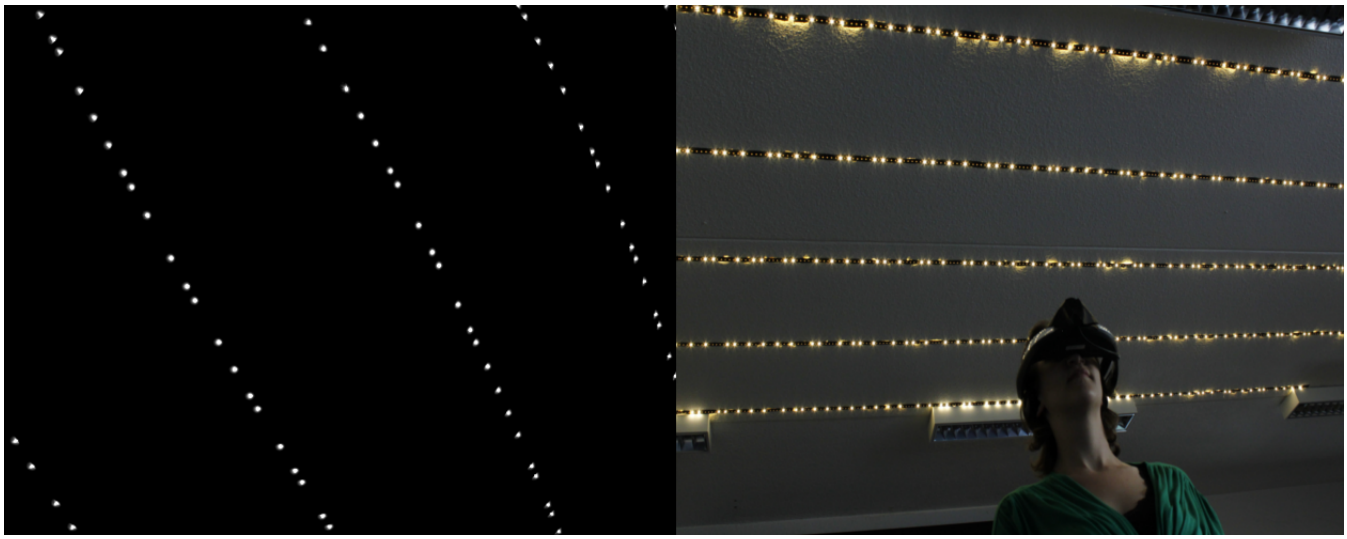

Figure 1: Prototype of our scalable optical tracking system. Left: Image taken from the camera used for tracking. Right: Overview of our system with the camera mounted on a head mounted display.

when navigating through a large environment. This advantage of having a limited distance, independent of the scale of the environment, means that we have no loss of accuracy when scaling up the system in comparison to most tracking systems. To achieve a global positioning method we need to encode the global position in the markers. Rather than using time multiplexing like the HiBall system [23] or printed patterns like the VisTracker [27], we chose to encode it spatially in relation to it's neighboring markers on a line, represented by the on and off pattern of the lights. The pattern detected by the camera and the overall setup are shown in Figure 1.

3.1 Encoded Lines

The encoding consists of a collection of parallel lines, with the position in the line as the u coordinate, and the actual line as the v coordinate. To encode the global position of each marker spatially, we propose to encode the marker pattern (i.e. the on and off pattern of the lights) per line of the grid in the u dimension using the De Bruijn sequence [5]. The De Bruijn sequence is a 1D cyclic sequence of a given alphabet (in our case 0,1) where every subsequence of a predefined length n appears exactly once. For our tracking system this means that when observing any part of the encoded pattern, we can exactly determine where in the sequence this pattern originated. This gives us a 1D unique position in a pattern of total size 2^n. An example of the pattern is shown in Appendix A.

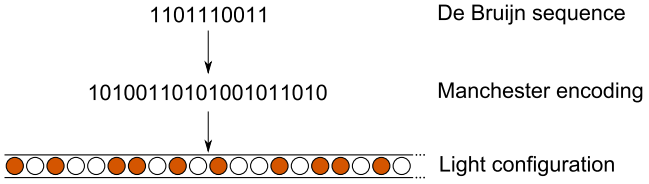

Figure 2: Overview of the encoding for one line. The De Bruijn code is first transformed to its Manchester encoding, and is subsequently directly translated to on and off lights.

The De Bruijn sequence has some extensions to the 2D domain, but this would require a substantial amount of markers. Instead, we propose a different coding system in the v direction to acquire 2D location. Since the De Bruijn Sequence is a cyclic sequence, it does not matter where in the sequence the pattern starts. Therefore we propose to encode the position in the v direction as a unique shift in adjacent sequences. So a 2D global position can be recovered by observing 2 parallel encoded lines. The space between the lights on parallel lines in the u dimension is much smaller than the distance between the actual lines (3 cm and 50 cm respectively). This will allow the distinction between the u and v direction.

As with all spatial markers, a minimal area of the pattern needs to be observed. In our case, we would assume that the camera always observes 1 square meter of ceiling. This means that the distance between each line needs to be at most 0.5 meter. The size of the working area of the tracker will only be defined by the number of markers visible in 1 meter, namely n. The maximum dimensions of the working area of the tracker are easily calculated:

$$\begin{aligned} dimension_u &= 2^n/n \ \text{ meters} \\ dimension_v &= 2^n/2 \ \text{ meters} \end{aligned} \tag{1}$$

We can see that for $n > 2$ the working area is not a square. To make it so, we can limit the range of shifts to be used. This makes it also more robust to detect a valid shift. Using this encoding system, we can see that the working area of the tracking system grows exponential in relation to the number of markers visible. For example when using a 8 bit pattern, more than 1 square kilometers can be encoded uniquely. When 15 markers are visible at all times, the pattern will be as large as 4.8 million km^2, or half the size of the United States.

2 bits	Encoded bits	Manchester distance
0 0	0 **1** 0 **1**	2 d
0 1	0 **1** **1** 0	1 d
1 0	**1** 0 0 **1**	3 d
1 1	**1** 0 **1** 0	2 d

Table 1: All the possible values for the Manchester distance. The patterns are encoded with the Manchester encoding, ensuring exactly two visible lights in the encoded pattern per two bits. The Manchester distance is then the distance between these two lights, where d is the distance between lights (on or off). These distances can be used to decode the visible light pattern.

To ensure enough markers are visible for tracking, we suggest to use a Manchester Encoding per bit, where two lights (on or off) are required for the encoding of one bit of the De Bruijn code. This is depicted in Figure 2. This means that when a bit has a value of 1, the lights are encoded as 10; when it has has the value 0, the lights are encoded as 01. Considering two adjacent bits, we can calculate the 'Manchester distance' between 2 adjacent active markers, i.e. visible lights, as can be seen in Table 1.

where $d = \frac{1}{2n}$ meter. This distance will become important when decoding the pattern.

Using this encoding, for $n = 15$, every bit of the De Bruijn code is encoded using two lights (on or off), and every unique location requires 15 bits (i.e. 30 lights). To acquire a 2D location, i.e. (u, v) coordinates, at least two lines are required, resulting in 60 lights per unique 2D location in the pattern.

3.2 Decoding Pattern

Our tracking system uses a camera to observe the encoded ceiling, as can be seen in Figure 1. After determining the image coordinates of the visible markers (i.e. visible lights), the pattern needs to be decoded to identify the marker identifiers, i.e. where the markers are located in the De Bruijn sequence. This is again represented by (u, v) coordinates. Because the dimmed lights are not visible, we use the distance between visible lights. The process is depicted in Figure 3.

First, we make a distinction between lines. The space between lines is much larger than the space between points on the lines. This allows to determine the lines to decode the patterns on. Once we have determined the lines, we will decode the lights on two consecutive lines to determine the marker identifiers on these lines.

Important to notice is that under projective transformation, as is the case with standard cameras, distance and the relative distances are not preserved [8]. However, we can see that in our design of the pattern, the bits of a De Bruijn sequence will be collinear. Therefore, we can decode the pattern on a line using the cross-ratio Ψ of collinear marker points, which is projective invariant. The cross-ratio Ψ of 4 adjacent collinear points p_1, p_2, p_3 and p_4 and distances $d_1 = ||p_2 - p_1||$, $d_2 = ||p_3 - p_2||$ and $d_3 = ||p_4 - p_3||$ can be calculated as follows:

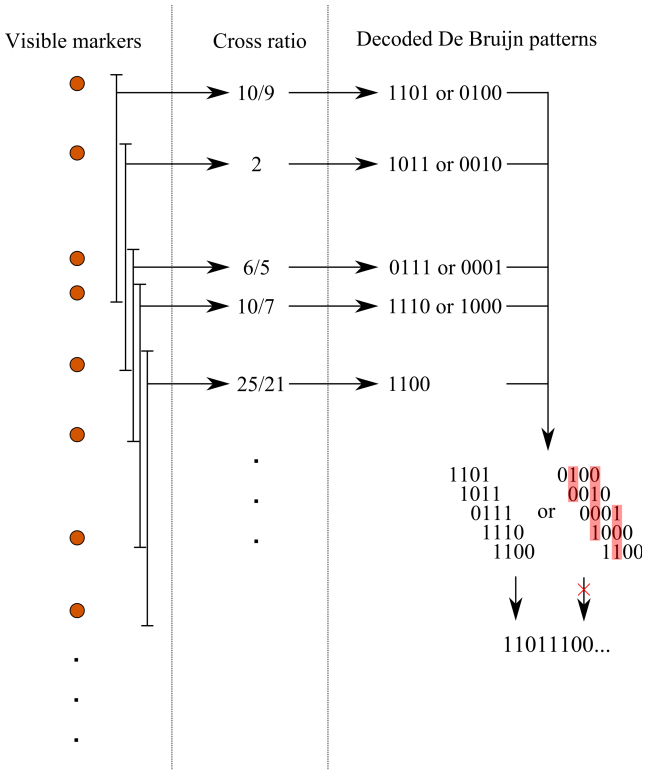

Figure 3: Overview of the decoding phase for one line. First, the visible markers are detected. Next, the distance for four consecutive visible markers is used to calculate the cross-ratio, which are used to acquire a partial De Bruijn sequence of 4 bits. Lastly, the partial sequences are combined to one sequence of 15 bits by overlapping the partial sequences. The left combination is read from left to right; the right combination from right to left. For some cross-ratios, multiple reading directions are possible. However, they will not match in a 15 bit pattern, as demonstrated at the right. The final combined sequence has a unique location in the complete De Bruijn sequence.

$$\Psi(p_1, p_2, p_3, p_4) = \frac{(d_1 + d_2)(d_2 + d_3)}{d_2(d_1 + d_2 + d_3)} \tag{2}$$

Because of the Manchester encoding, every four subsequent visible lights corresponds with 4 bits in the De Bruijn code, where the code is determined by the distance between the visible lights. Using the knowledge of the 'Manchester distances' (Table 1), we can show that there are only 10 valid cross-ratios in a pattern and each is perspective invariant. We use these ratios to decode the index id_p of each point p in the De Bruijn sequence of a line. Table 2 gives the different cross-ratios and their corresponding patterns.

For $n = 15$, we need to decode 11 subsequent and overlapping sets of four visible lights. This way, 11 overlapping codes can be obtained, and thus a 15 bit De Bruijn code is acquired. Looking up this 15-bit code gives us the index in

$$\begin{bmatrix} 0 & 0 & 0 & -X_{p_1} & -Z_{p_1} & -1 & y_{p_1}X_{p_1} & y_{p_1}Z_{p_1} & y_{p_1} \\ X_{p_1} & Z_{p_1} & 1 & 0 & 0 & 0 & -x_{p_1}X_{p_1} & -x_{p_1}Z_{p_1} & -x_{p_1} \\ -y_{p_1}X_{p_1} & -y_{p_1}Z_{p_1} & -y_{p_1} & x_{p_1}X_{p_1} & x_{p_1}Z_{p_1} & x_{p_1} & 0 & 0 & 0 \\ 0 & 0 & 0 & -X_{p_2} & -Z_{p_2} & -1 & y_{p_2}X_{p_2} & y_{p_2}Z_{p_2} & y_{p_2} \\ X_{p_2} & Z_{p_2} & 1 & 0 & 0 & 0 & -x_{p_2}X_{p_2} & -x_{p_2}Z_{p_2} & -x_{p_2} \\ -y_{p_2}X_{p_2} & -y_{p_2}Z_{p_2} & -y_{p_2} & x_{p_2}X_{p_2} & x_{p_2}Z_{p_2} & x_{p_1} & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \times \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \quad (3)$$

Figure 4: The full set of linear equations for two points for estimating the camera pose using 2D-3D correspondences.

Pattern	Ψ	Pattern	Ψ
0 0 0 0	4/3	1 0 0 0	10/7
0 0 0 1	6/5	1 0 0 1	5/4
0 0 1 0	2	1 0 1 0	16/7
0 0 1 1	9/5	1 0 1 1	2
0 1 0 0	10/9	1 1 0 0	25/21
0 1 0 1	16/15	1 1 0 1	10/9
0 1 1 0	5/4	1 1 1 0	10/7
0 1 1 1	6/5	1 1 1 1	4/3

Table 2: Different De Bruijn patterns of 4 bits and their corresponding cross-ratios. By calculating the cross-ratio of 4 visible lights, which are Manchester encoded patterns, its pattern can be decoded.

the complete De Bruijn sequence, denoted as id_p.

However, as can be seen, there are 10 different cross-ratios Ψ for 16 codes. While this introduces an ambiguity for four points, this ambiguity is practically eliminated when using 11 cross-ratios, i.e. 15 points, or more. The ambiguity for four points is caused by the direction the Manchester encoded pattern is read. For example, the pattern 0100 is Manchester encoded as 01100101 and the pattern 1101 is encoded as 10100110. As can be seen, these Manchester encoded patterns are equal when one is reversed. Therefore, we read both directions and try to match the complete pattern in both directions. One of the directions is not valid if the code contains a non-ambiguous cross-ratio, resulting in one valid reading direction. In the other non-valid direction, partial patterns of 4 bits will not overlap correctly.

We will decode two adjacent lines to determine the v coordinate we are processing. In our setup, 2 adjacent pattern lines have a unique shift s. The De Bruijn code is known for the two lines, allowing the determination of this shift of two n-bit patterns easily by looking up the two codes in the complete sequence.

Finally, identifying the line- and marker-id of each marker p can be done as follows:

$$\begin{cases} v_p = MOD(s, 2^n) \\ u_p = id_p - MOD(v_p(v_p+1)/2, 2^n) \end{cases} \quad (4)$$

As we now know the unique identifier of each marker p, we can determine the accompanying 3D coordinates. We know the height of the ceiling, the distance between the led markers and the distance between the lines. Using this information, transforming marker coordinates (u, v) to 3D world coordinates is straightforward:

$$\begin{cases} X = 2du_p + (1 - DB(p))d \\ Z = v_p\Delta_Y \end{cases} \quad (5)$$

where the plane XZ lies parallel to the ceiling, d is the distance between the markers (on or off) in the u direction and Δ_Y is the distance between the lines. The function $DB(p)$ determines the De Bruijn bit (0 or 1) of marker p, adding d to X if the code is 0, compensating for the Manchester encoding. In out setup $\Delta_Y = 0.5m$ and $d = 1/2n = 3cm$. We assume $Y = 0$. Now we have a set of 2D image coordinates of the markers, together with the corresponding decoded 3D coordinates. This set of 2D-3D correspondences will now be used for the estimation of the camera pose.

3.3 Estimating Camera Pose

From the identification of the markers, we got a set of 2D-3D correspondences. The relation between those correspondences is defined by the standard pinhole camera model:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = K.[R|R.T]. \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} \quad (6)$$

where $[XYZW]^T$ are the homogeneous coordinates of a 3D point in the world and $[xyw]^T$ its projection in image space. K contains the intrinsic camera parameters (focal length, principal point, ...) which we will be considering fixed and known after standard intrinsic calibration [8]. The 3x3 matrix R and vector T are the extrinsic parameters rotation and translation which will be the result of our tracking algorithm, fully determining the pose and location of the camera.

Since all points lie on a plane (the ceiling) with $Y = 0$, a homography can be calculated to have a first estimation of the pose of the camera. Given the standard pinhole camera equation (Eq. 6), we can see that for each homogeneous

Figure 5: The LED setup. Left: The LED strips used in our setup. Middle: LED strips mounted to the ceiling, displaying the pattern used for decoding. Right: Detailed view of encoded LED strip.

point correspondence $(x_p, y_p, 1)$ - $(X_p, Y_p, Z_p, 1)$ satisfies the following similarity (equal up to an unknown scale):

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \sim \begin{bmatrix} r_{11} & r_{13} & (R.T)_x \\ r_{21} & r_{23} & (R.T)_y \\ r_{31} & r_{33} & (R.T)_z \end{bmatrix} . \begin{bmatrix} X_p \\ Z_p \\ 1 \end{bmatrix} = H. \begin{bmatrix} X_p \\ Z_p \\ 1 \end{bmatrix} \tag{7}$$

where $(x_p, y_p, 1) = K^{-1}[imgP_x, imgP_y, 1]^T$ are the coordinates in camera space and $Y_p = 0$.

This 3x3 matrix defines a projective transformation known as a homography H. Using SVD (Singular Value Decomposition), this matrix can be calculated with at least 4 image correspondences by solving the set of linear equations, created from the correspondence points as knowns and the values of the homography as unknowns [8]. The full set of linear equations for two points is depicted in Equation 3.

Solving for H gives us the vectors $[r_{11}r_{21}r_{31}]^T$ and $[r_{13}r_{23}r_{33}]^T$ up to an unknown scale. But both vectors should have had length 1 and be orthogonal because they are the basis of the camera coordinate system. So we can correct for that. The second column of the rotation matrix R, i.e. the third base vector, can be calculated as follows:

$$\begin{bmatrix} r_{12}r_{22}r_{32} \end{bmatrix} = \begin{bmatrix} r_{11}r_{21}r_{31} \end{bmatrix} \times \begin{bmatrix} r_{13}r_{23}r_{33} \end{bmatrix} \tag{8}$$

Given the estimate of the rotation matrix R, translation T can be calculated from the third column of the homography.

3.4 Refining Camera Pose

The calculation of the extrinsic camera parameters using a linear homography gives us a good estimation of the camera pose. However, this method is highly sensitive to errors in the input data. We therefore wish to refine this estimate using non-linear optimization.

First of all, we eliminate outliers of the current camera estimation using a RANSAC approach [6]. Outliers include other light sources or noise in the image. This leaves us with good 2D-3D correspondences to work with. We propose to minimize the reprojection error of the known world pattern.

Figure 6: The Sony HMZ-T1 head mounted display, which was used in our prototype to allow the user to navigate a virtual world.

$$min_{R,T}(\sum_{i=1}^{m} || \begin{bmatrix} x_i \\ y_i \\ 1 \\ 1 \end{bmatrix} - K.[R|R.T]. \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} ||) \tag{9}$$

We minimize this function using the Levenberg-Marquardt algorithm [11]. The result is an improved rotation and translation matrix representing the current camera pose.

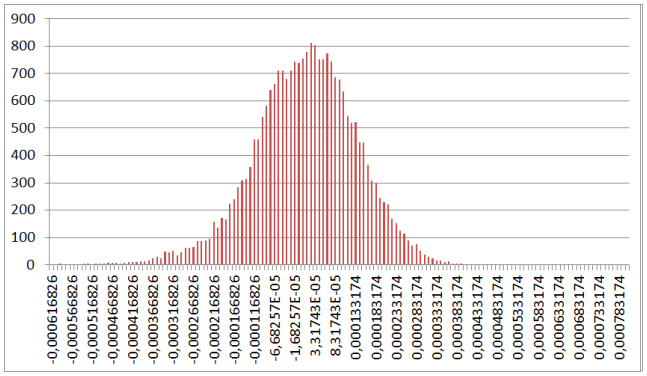

Figure 7: The jitter distribution for the yaw turning direction.

4. OUR PROTOTYPE

To test our proposed scalable tracking system, a prototype setup was constructed in our lab. We propose to use LEDs because it has the advantage of being independent of environment lighting. It also reduces the effects of motion blur as the shutter time can be really short. We used readily available white LED strips to encode our De Bruijn sequence, but it can be replaced with infra-red LEDs if needed. LED strips already provide a uniform distance between individual LEDs. The setup comprised of 10 encoded lines of 5 meter each, which gave us about 25 m^2 of tracking space to test our approach. The LED strips contained 60 3528SMD LEDs/meter with each LED giving about 5 lumen of light over a field of view of 120°. We used tape to mask the markers that were inactive in the coding and reduced it to a 15 bit code as it was more than enough for our setup. The LED strips can be seen in Figure 5. Constructing this prototype costs us less than 500 EUR for a 25 m^2 tracking system (less than 20 EUR/m^2) by using only off-the-shelf hardware. This makes the system really cost-effective when constructing large installations.

We used a 'Point Grey Firefly MV' monochrome camera which can provide 752x480 images at 60 fps over a USB 2.0 connection. This computer vision camera is on the market for 200 EUR. The images were processed by a Intel i7 quad core processor with 4 GB of RAM. We provided the user with a Sony HMZ-T1 head mounted display (HMD) to navigate a virtual environment (see Figure 6). The camera was mounted on top of the HMD, as can be seen in Figure 1, to track the user's location and orientation in the environment. When adding more users, only an extra camera is required because the tracking system is independent of the number of users walking around.

5. RESULTS

To evaluate the performance of our tracking system, we constructed a real scene to analyze real errors. We did not perform any filtering or smoothing on these results to demonstrate the effectiveness of the method itself.

5.1 Jitter

We placed the camera on a static place and gathered pose data from 1000 frames. This allows us to analyze the jitter on real captured data. The following table describes the jitter. The distribution is similar for all measured degrees of freedom (3 for position, 3 for orientation). Figure 7 shows that the jitter for the yaw rotation is Gaussian distributed. Table 5.1 gives the actual numerical analysis.

Direction	Average Abs Difference	Standard Deviation
yaw	0.0059 degrees	0.0083
pitch	0.035 degrees	0.049
roll	0.092 degrees	0.12
x	0.001m	0.0015
y	0.0003m	0.00053
z	0.002m	0.0039

Table 3: Different values for the measured jitter, with standard deviation

As can be seen, the jitter is very small, only 2 millimeter in the Z direction and only 0.09 degrees in the roll.

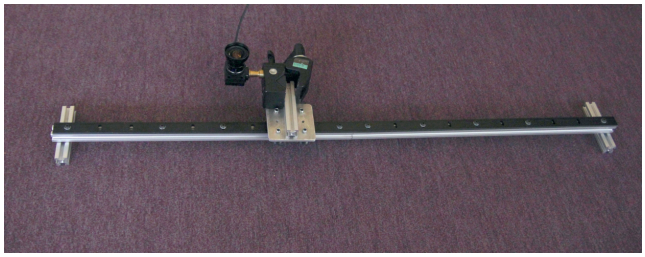

Figure 8: Setup for the straight line test. the camera movement is limited to one dimension.

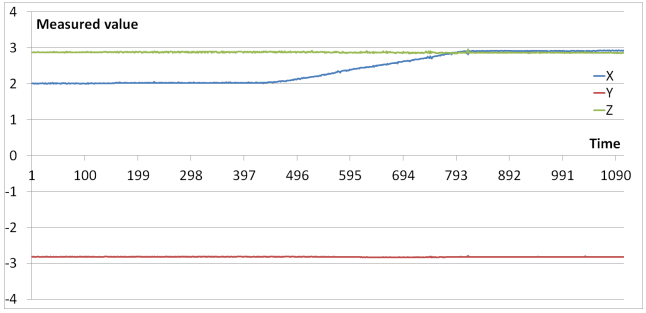

Figure 9: Values after the movement of the camera in the X direction. The movement is clearly visible, while the other directions are stable.

5.2 Movement in a Straight Line

We placed the camera on a fixed rail of one meter to assess the accuracy per direction (see Figure 8). The recovered positions are shown in Figures 9, 10, and 11. As can be seen, a straight walk is clearly visible, both when the rail was places aligned with the X axis (moving forward), aligned with the Y axis (moving up and down), and aligned with the Z axis (moving left). The varying direction is showing a distinct and constant movement, while the other directions show stable values. This results demonstrate the usefulness of the method for tracking global location. Occasional spikes can be perceived; typical spikes range around 5mm, as can be seen in Figure 9. These can be diminished by applying local filtering.

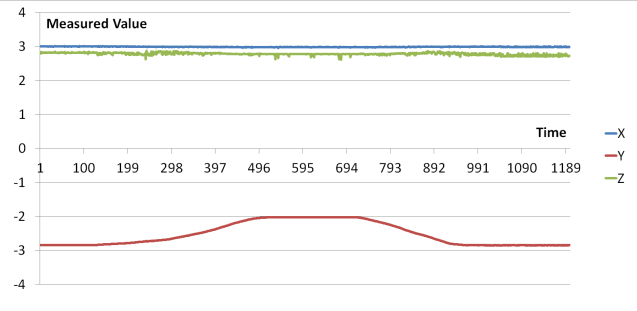

Figure 10: Values after the movement of the camera in the Y direction. The movement is clearly visible, while the other directions are stable.

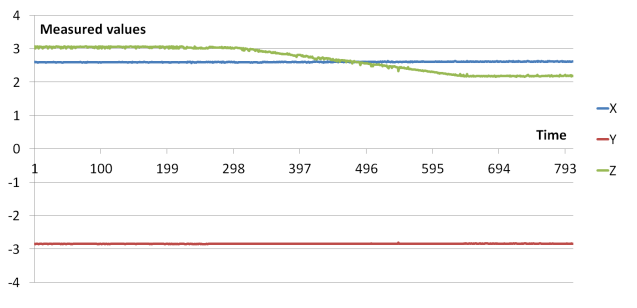

Figure 11: Values after the movement of the camera in the Z direction. The movement is clearly visible, while the other directions are stable.

Figure 12: Setup for the rotating test. the camera movement is limited to one turning direction.

5.3 Turning table

Finally, we placed the camera on a turning table to simulate uniform rotational movement (see Figure 12). Figure 13 shows the orientation results, represented by yaw, pitch, and roll. As can be seen, the pitch and roll are stable, while the yaw shows the turning of the table. Occasional spikes can be perceived; typical spikes range around 0.1 degrees, as can be seen in Figure 13. These can be diminished by applying local filtering.

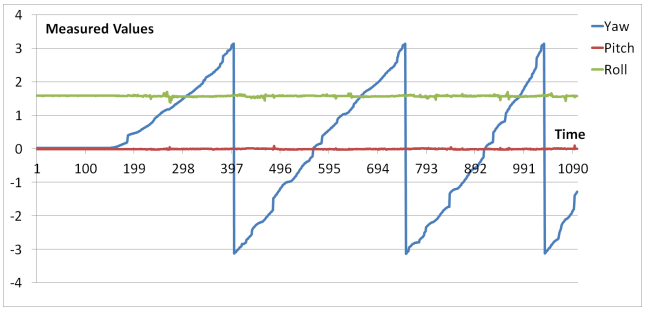

Figure 13: Values after the rotation of the camera on a turn table. The movement is clearly visible, while the other rotations are stable.

6. DISCUSSION

The results demonstrate that our method is accurate for navigating virtual environments in a large environment, using both global location and orientation. The jitter and error are small compared to other similar methods. Due to the design of the setup, no drift is possible. The software runs at 200 Hz, allowing a smooth and real-time interaction.

However, the method is limited by a few factors. Firstly, the camera should always see a part of the grid. If not, the tracking is lost. We plan to extend our system with inertial tracking methods to bridge those moments.

Secondly, the tracking accuracy and framerate is limited by the camera. The resolution determines the distinctiveness of the individual lights. If the resolution is too small, lights will blend and tracking will fail. Furthermore, the framerate of the system is limited by the camera framerate; in our system this limit is 60Hz, while the software can run at 200Hz.

Lastly, some jitter and outliers can be detected in the raw output. This can easily be solved with standard local filters, such as a Kalman filter [10] or a DESP filter [12]. However, filtering will introduce additional latency. In our system, we opted for a DESP filter. We did not show the filtered details to demonstrate the effectiveness of the system itself.

7. CONCLUSION

In this paper, we presented a novel optical tracking design for navigating large virtual environments. We proposed a spatial coding system of markers that is scalable both in terms of working area and number of users. The tracking system is designed to have a constant accurate result, no matter the dimensions of the environment, and gives an absolute position and orientation of the user. The results show the accuracy of the method.

A prototype of a 25 m^2 tracking system was built in our lab to validate the design. We also showed that building our tracking system for larger installations can be cost-effective. Adding a square meter of working area costs less than 20 EUR and adding another user adds 200 EUR to the overall cost. This is much cheaper than any comparable system currently on the market, while delivering similar tracking performance and accuracy.

8. REFERENCES

[1] T. G. Bishop. *Self-tracker: a smart optical sensor on silicon (vlsi, graphics)*. PhD thesis, 1984. AAI8415794.

[2] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. In *Proceedings of the IEEE Virtual Reality 2008*, pages 137–144, 2008.

[3] J. Cadman. Deploying commercial location-aware systems. In *Proceedings of the 2003 Workshop on Location-Aware Computing (held as part of UbiComp 2003)*, pages 4–6, 2003.

[4] S. S. Chance, F. Gaunet, A. C. Beall, and J. M. Loomis. Locomotion mode affects the updating of objects encountered during travel: The contribution of vestibular and proprioceptive inputs to path integration. *Presence: Teleoper. Virtual Environ.*, 7(2):168–178, Apr. 1998.

[5] N. G. De Bruijn. A combinatorial problem. *Koninklijke Nederlandse Akademie v. Wetenschappen*, 49:758–764, 1946.

[6] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[7] E. Foxlin and L. Naimark. Vis-tracker: A wearable vision-inertial self-tracker. In *Proceedings of the IEEE Virtual Reality 2003*, VR '03, pages 199–, Washington, DC, USA, 2003. IEEE Computer Society.

[8] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[9] H. C. J. Hofmann-Wellenhof, B.; Lichtenegger. Global positioning system. theory and practice. 1993.

[10] R. E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[11] C. Kelley. *Iterative Methods for Optimization*. Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, 1987.

[12] J. J. LaViola. Double exponential smoothing: an alternative to kalman filter-based predictive tracking. In *Proceedings of the workshop on Virtual environments 2003*, pages 199–206. ACM, 2003.

[13] S. Maesen and P. Bekaert. Scalable optical tracking - a practical low-cost solution for large virtual environments. In *VISAPP 2011 - Proceedings of the Sixth International Conference on Computer Vision Theory and Applications*, pages 538–545, 2011.

[14] C. T. Neth, J. L. Souman, D. Engel, U. Kloos, H. H. BuÂĺthoff, and B. J. Mohler. Velocity-dependent dynamic curvature gain for redirected walking. In *Proceedings of the 2011 IEEE Virtual Reality Conference*, VR '11, pages 151–158, Washington, DC, USA, 2011. IEEE Computer Society.

[15] T. C. Peck, H. Fuchs, and M. C. Whitton. Improved redirection with distractors: A large-scale-real-walking locomotion interface and its effect on navigation in virtual environments. In *Proceedings of the 2010 IEEE Virtual Reality Conference*, VR '10, pages 35–38, Washington, DC, USA, 2010. IEEE Computer Society.

[16] T. Pintaric and H. Kaufmann. Affordable infrared-optical pose-tracking for virtual and augmented realityr. In *Proceedings of Trends and Issues in Tracking for Virtual Environments Workshop, IEEE VR 2007*. Shaker-Verlag, 2007.

[17] R. Raskar, H. Nii, B. Dedecker, Y. Hashimoto, J. Summet, D. Moore, Y. Zhao, J. Westhues, P. Dietz, J. Barnwell, et al. Prakash: lighting aware motion capture using photosensing markers and multiplexed illuminators. In *ACM Transactions on Graphics (TOG)*, volume 26, page 36. ACM, 2007.

[18] R. A. Ruddle and S. Lessels. The benefits of using a walking interface to navigate virtual environments. *ACM Trans. Comput.-Hum. Interact.*, 16(1):5:1–5:18, Apr. 2009.

[19] E. Suma, Z. Lipps, S. Finklestein, D. M. Krum, and M. Bolas. Impossible spaces: Maximizing natural walking in virtual environments with self-overlapping architecture. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):555–564, Apr. 2012.

[20] E. A. Suma, S. L. Finkelstein, M. Reid, S. V. Babu, A. C. Ulinski, and L. F. Hodges. Evaluation of the cognitive effects of travel technique in complex real and virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 16:690–702, 2010.

[21] I. E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68 (Fall, part I), pages 757–764, New York, NY, USA, 1968. ACM.

[22] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F. P. Brooks, Jr. Walking > walking-in-place > flying, in virtual environments. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 359–364, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[23] G. Welch, G. Bishop, L. Vicci, S. Brumback, K. Keller, and D. Colucci. High-performance wide-area optical tracking: The hiball tracking system. *Presence: Teleoper. Virtual Environ.*, 10(1):1–21, Feb. 2001.

[24] G. F. Welch. Scaat: Incremental tracking with incomplete information. Technical report, Chapel Hill, NC, USA, 1996.

[25] B. Williams, G. Narasimham, B. Rump, T. P. McNamara, T. H. Carr, J. Rieser, and B. Bodenheimer. Exploring large virtual environments with an hmd when physical space is limited. In *Proceedings of the 4th symposium on Applied perception in graphics and visualization*, APGV '07, pages 41–48, New York, NY, USA, 2007. ACM.

[26] M. G. Wing, A. Eklund, and L. D. Kellogg. Consumer-grade global positioning system (gps) accuracy and reliability. *Journal of Forestry*, 103(4):169–173, 2005.

[27] D. Wormell, E. Foxlin, and P. Katzman. Advanced Inertial-Optical Tracking System for Wide Area Mixed and Augmented Reality Systems . pages 65–68, Weimar, Germany, 2007. Eurographics Association.

APPENDIX
A. EXAMPLE OF THE DE BRUIJN CODE

Here, we give a part of the complete De Bruijn sequence. Every 15 consecutive bits only appear once in the code, allowing the mapping between a 15-bit code and a location in the sequence. The code is constructed by generating a new bit to the end of an existing 15-bit code. The new, 16th bit is the result of the `xor` operation of the first two bits. The code is then again reduced to 15 bits by dropping the first code. This method will generate a code where every subcode of 15 bits is unique in the complete code [5]. The code is cyclic, thus every 15-bit code can be used as starting code.

. . . 1 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 0 1 1 0 1 0 1 1 0 0 1 1 1 0
1 1 0 1 1 1 1 0 1 0 1 0 0 1 1 0 1 1 0 0 0 1 1 1 1 1 0 1 0 1 1 0
1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 1 0 1 1 0 0 0 1 0 0 0 1 1 0 0 1 1
0 1 0 0 1 1 0 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 1 1 1 1 1 1 0 0 1
1 1 1 1 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0
0 1 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 1 1 0 0 1 1 0 0 1 1 1
1 0 0 1 0 1 0 1 0 1 0 1 0 0 0 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1
0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 0
0 0 0 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 0 1 1 0 0 1 0 1 1 1 0 0 0 0
0 1 0 1 0 1 1 1 0 0 1 0 0 0 0 1 1 1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0
0 0 1 0 1 1 1 0 1 0 0 1 1 0 0 0 1 1 1 0 0 1 1 1 0 1 0 1 0 0 1 0
0 1 0 1 0 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 0 1 0 0 0 0 1 1 0 1 1 . . .